DIRECTION-PRESERVING CREDIT REDISTRIBUTION FOR LONG-HORIZON LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon LLM agents must infer which individual actions deserve credit from sparse trajectory-level outcomes. Existing step-level estimators provide finer-grained supervision, yet the resulting credit is not uniformly reliable: *which parts of these signals should the policy trust?* We reveal a complementary asymmetry between non-parametric and parametric credit. Non-parametric advantages, derived from realized-return contrasts, provide informative update directions, but their magnitudes remain coupled to sampled continuations and can misallocate credit across actions. Conversely, accurate value or reward estimation is difficult in long-horizon agentic tasks, causing parametric prediction errors to propagate into policy advantages and even reverse update directions; nevertheless, learned value movement retains useful information about relative action contribution. Based on this observation, we introduce Grounded Relative-Value Advantage Weighting (GRAW), a plug-in for non-parametric step-level estimators. GRAW retains the base credit as a grounded allocation prior and uses learned relative value movement only to reweight actions within the same update direction, preserving every base advantage sign and the total absolute credit in each direction. Across three long-horizon agent benchmarks and multiple model scales, GRAW consistently improves strong non-parametric step-level baselines, achieves the best overall performance, reduces variance across seeds, and adds only minimal computational overhead.

1 INTRODUCTION

Reinforcement learning (RL) is increasingly used to train large language models (LLMs) as agents that act over extended horizons: browsing the web (Gandhi & Neubig, 2026; Hsu et al., 2026), operating tools and executable environments (He et al., 2026b; Shao et al., 2026), and acting in embodied worlds (Choi et al., 2024; Parashar et al., 2026). In many of these settings, supervision arrives only after a long sequence of decisions. A terminal outcome tells the agent *what* happened, but not which of the preceding actions made the difference. Learning therefore hinges on *credit assignment* (Zhang, 2026): determining which actions should be reinforced or suppressed and how strongly they should influence the policy update.

Recent non-parametric methods (Feng et al., 2025; He et al., 2026a) improve credit granularity by contrasting realized returns following alternative actions from the same or comparable states. Yet finer-grained credit does not necessarily mean that actions are weighted according to their actual contribution. Each sampled action is still evaluated through a single continuation, so its credit reflects not only the action itself but also the downstream decisions that happen to follow it. Consequently, a routine action and a decisive action can receive similar credit despite contributing very differently to the final outcome. Parametric approaches offer an alternative by learning state- or action-specific signals from accumulated experience, using explicit value critics (Schulman et al., 2017) or learned step-reward models (Cui et al., 2025; Liu et al., 2026). However, accurate value or reward estimation is difficult in long-horizon agentic tasks, and prediction errors can propagate directly into policy credit and even reverse its update direction. This suggests a sharper question: *if neither form of credit is reliable in full, which part of each signal should the policy trust?*

Our analysis (Section 3) reveals a complementary asymmetry. Although non-parametric credit remains coupled to sampled continuations, its advantage signs still provide informative update directions. Conversely, although parametric estimators can produce incorrect advantage signs due

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

to value or reward estimation errors over long horizons, learned value movement retains useful information about relative action quality. The key is therefore not to replace one estimator with the other, but to restrict each signal to the role it supports reliably. Based on this principle, we introduce **G**rounded **R**elative-Value **A**dvantage **W**eighting (**GRAW**), a plug-in framework for non-parametric step-level estimators that preserves their grounded credit structure while restricting learned relative value movement to within-direction reweighting.

Specifically, GRAW attaches a lightweight value head to detached policy representations and trains it using outcome-derived targets from the same rollouts used for policy optimization. Changes in its predictions across consecutive states provide a relative signal for reweighting the base advantages. Positive and negative credit are redistributed separately within each trajectory: strictly positive multipliers preserve every base advantage sign, while normalization preserves the total absolute credit in each direction. Thus, GRAW combines outcome-grounded non-parametric credit with relative information from learned values without allowing learned predictions to redefine the underlying credit. Because the head reuses policy hidden states, GRAW requires no additional backbone forward passes, rollouts, or environment interactions. Across WebShop (Yao et al., 2022), SciWorld (Wang et al., 2022), and ALFWorld (Shridhar et al., 2021), GRAW consistently improves both GiGPO and HGPO, reduces variance across seeds, and maintains its gains from 1.5B to 7B parameters with only minor computational overhead.

Our contributions are threefold: (1) We reveal a complementary asymmetry in step-level credit: non-parametric advantages retain informative update directions despite continuation-dependent magnitudes, whereas learned value movement retains useful relative action information despite unreliable direct policy credit. (2) We introduce GRAW, a plug-in framework that uses learned relative value movement to reweight non-parametric credit within each trajectory while preserving every base advantage sign and the total absolute credit in each direction. (3) Through extensive experiments and analyses, we show that GRAW consistently improves strong step-level baselines across model scales, reduces variance across seeds, produces credit reweighting aligned with relative action contribution, and largely preserves the computational efficiency of non-parametric RL.

2 PRELIMINARIES

Agentic reinforcement learning. We consider an LLM agent that interacts with an environment to complete a multi-step task $x \sim p(\mathcal{X})$. At step t, the agent observes a state s_t and samples a textual action $a_t \sim \pi_\theta(\cdot \mid s_t, x)$. The environment returns a scalar reward $r_t \in \mathbb{R}$ and transitions to s_{t+1}, producing a trajectory $\tau = ((s_1, a_1, r_1), \ldots, (s_T, a_T, r_T))$. We focus on sparse terminal supervision, where $r_t = 0$ for $t < T$ and $r_T = R \in [0, 1]$. The realized return-to-go is

$$G_t = \sum_{k=t}^{T} \gamma^{k-t} r_k = \gamma^{T-t} R, \tag{1}$$

where $\gamma \in (0, 1]$ is the discount factor. For a fixed task x, the policy action value, state value, and advantage are defined as (Sutton et al., 1998)

$$\begin{aligned} &Q^\pi(s_t, a_t) = \mathbb{E}[G_t \mid s_t, a_t, x], \qquad V^\pi(s_t) = \mathbb{E}_{a\sim\pi}[Q^\pi(s_t, a)], \\ &A^\pi(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t), \end{aligned} \tag{2}$$

where task dependence is suppressed in the value-function notation. In practice, RL algorithms construct an estimator-specific advantage A_t to approximate A^π; we refer to A_t as *policy credit*. Omitting algorithm-specific importance weighting, clipping, and regularization, its contribution to the policy update can be written schematically as

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau\sim\pi_\theta}\Big[\sum_{t=1}^{T} A_t \, \nabla_\theta \log \pi_\theta(a_t \mid s_t, x)\Big]. \tag{3}$$

Here, the sign of A_t specifies whether the sampled action is reinforced or suppressed, while $|A_t|$ weights its gradient contribution. This distinction allows us to separately examine the sign of policy credit and how its magnitude is distributed across actions.

Non-parametric step-level credit. For each task x, group-relative RL samples N trajectories $\{\tau^{(i)}\}_{i=1}^{N}$ and constructs advantages by comparing their realized outcomes, without a learned credit

predictor (Shao et al., 2024; Ahmadian et al., 2024). Step-level methods such as GiGPO (Feng et al., 2025) and HGPO (He et al., 2026a) extend these comparisons to actions taken from the same state. We denote a non-parametric step-level base estimator by $\mathcal{B}$ and write its credit for step t of trajectory i as

$$A^{\mathcal{B}}_{i,t} = \lambda_E A^E_i + \sum_{k=0}^{K_{\text{ctx}}} w_k \, A^{S,k}_{i,t}, \tag{4}$$

where A^E_i is the trajectory-level empirical credit shared by all actions of $\tau^{(i)}$, $A^{S,k}_{i,t}$ is the step-level empirical credit at context depth k, w_k weights that depth, and K_{ctx} is the maximum context depth considered by the estimator. Let $\mathcal{C}_k(s_{i,t})$ index the sampled state–action pairs in the rollout group whose current state and k preceding states coincide with $(s_{i,t-k}, \ldots, s_{i,t})$. The step-level credit contrasts the realized return of $a_{i,t}$ against this group,

$$A^{S,k}_{i,t} = \frac{G_{i,t} - \mu\big(\mathcal{C}_k(s_{i,t})\big)}{F_{\text{norm}}\big(\mathcal{C}_k(s_{i,t})\big)}, \tag{5}$$

where $\mu(\mathcal{C}_k(s_{i,t}))$ is the mean realized return over the group $\mathcal{C}_k(s_{i,t})$ and $F_{\text{norm}}(\mathcal{C}_k(s_{i,t}))$ is the estimator-specific normalization, either the standard deviation or 1. The trajectory-level credit A^E_i is computed analogously from terminal outcomes $\{R^{(j)}\}_{j=1}^N$. GiGPO uses $\lambda_E = 1$ with only the anchor-state group ($K_{\text{ctx}} = 0, w_0 = 1$), whereas HGPO uses $\lambda_E = 0$ and $w_k \propto (k+1)^\alpha$, normalized over valid context depths. Although the comparison baseline pools multiple returns, $G_{i,t}$ comes from a single observed continuation rather than the expectation defining $Q^\pi(s_{i,t}, a_{i,t})$. We omit i when unambiguous hereafter.

Parametric step-level credit. Parametric methods learn state- or action-specific predictions from accumulated experience and use them to construct policy credit. We consider two representative forms. First, PPO (Schulman et al., 2017) uses a learned state-value critic $V_\phi(s_t)$ to construct advantages through generalized advantage estimation (GAE) (Schulman et al., 2016):

$$\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t), \qquad A^{\text{PPO}}_t = \sum_{\ell=0}^{T-t} (\gamma\lambda)^\ell \delta_{t+\ell}, \tag{6}$$

where $\lambda \in [0, 1]$ is the GAE parameter. Second, implicit step-reward methods such as PRIME (Cui et al., 2025) and iStar (Liu et al., 2026) learn intermediate reward signals from trajectory-level supervision and incorporate them into policy advantages without an explicit state-value critic. In both cases, learned predictions directly enter credit construction. Appendix A provides the detailed formulations.

3 Analysis – What Can We Trust in Step-Level Credit?

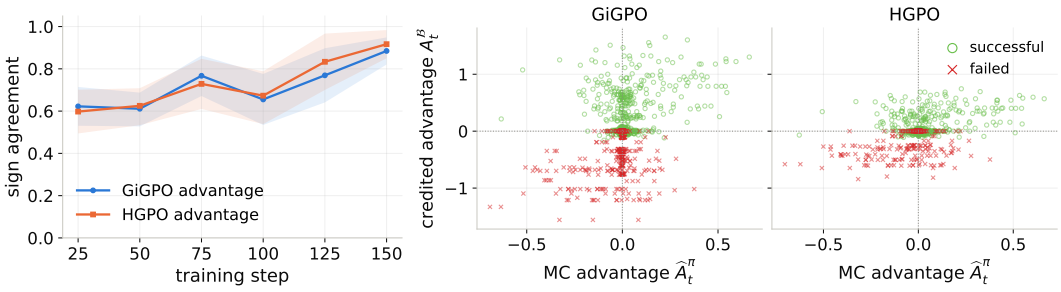
3.1 Non-Parametric Credit: Informative in Direction, Confounded by Sampled Continuations

We examine non-parametric step-level credit along two dimensions: whether its sign provides an informative update direction and whether its magnitude distinguishes relative action quality. We analyze the advantages actually used by the policy objectives of GiGPO (Feng et al., 2025) and HGPO (He et al., 2026a), denoted by $A^{\mathcal{B}}_t$. GiGPO combines trajectory- and step-level credit, whereas HGPO relies solely on hierarchical step-level contrasts, allowing us to examine both constructions. On WebShop (Yao et al., 2022), we compare these credits against a Monte-Carlo estimate of the advantage under the corresponding current policy:

$$\widehat{A}^\pi_t = \widehat{Q}^\pi(s_t, a_t) - \widehat{V}^\pi(s_t), \tag{7}$$

where $\widehat{Q}^\pi$ and $\widehat{V}^\pi$ are estimated from 32 independent continuations under the current policy.

Direction: realized-return contrasts provide informative policy guidance. We first compare the sign of $A^{\mathcal{B}}_t$ with that of the Monte-Carlo reference advantage. Figure 1a shows that sign agreement remains above chance throughout training for both GiGPO and HGPO, with higher agreement at later checkpoints. Thus, empirical return contrasts provide informative guidance about whether a sampled



(a) Sign agreement rate (b) Credit vs. Monte-Carlo advantage

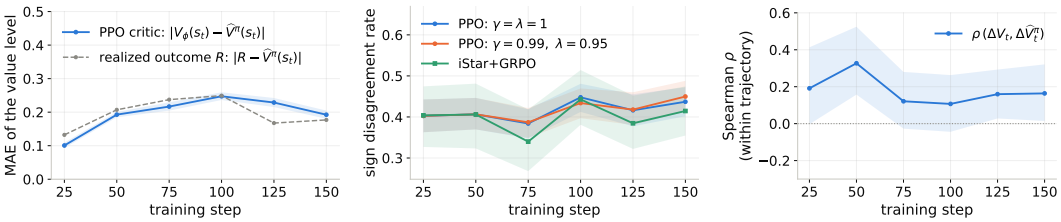
Figure 1: **Analysis of non-parametric step-level credit.** (a) Sign agreement rate between $A_t^{\mathcal{B}}$ and the Monte-Carlo reference $\widehat{A}_t^{\pi}$ across training; bands denote 95% bootstrap confidence intervals over trajectories. (b) Empirical credit $A_t^{\mathcal{B}}$ versus $\widehat{A}_t^{\pi}$, with each action colored by its trajectory outcome.

action should be reinforced or suppressed, supporting their use as an outcome-grounded directional anchor.

Allocation: credit magnitudes remain coupled to sampled outcomes. Figure 1b shows that actions with similar Monte-Carlo advantages can still receive markedly different empirical credit depending on the trajectory outcome. This dependence is more pronounced for GiGPO but persists in HGPO despite its exclusively step-level construction. The pattern is consistent with evaluating each action through a sampled continuation, whose return also reflects subsequent decisions. For example, within the same successful trajectory, a routine action with a small reference advantage and a decisive action with a much larger one can receive similarly strong positive credit. Thus, non-parametric credit is more reliable for update direction than for relative allocation.

3.2 Parametric Credit: Unreliable as Direct Credit, Useful for Relative Ranking

We examine whether learned predictions retain useful information for relative credit allocation. We analyze two representative estimators on WebShop: PPO's explicit value critic (Schulman et al., 2017) and iStar's implicit step-reward model (Liu et al., 2026). We compare PPO's value predictions $V_\phi(s_t)$ with the current-policy Monte-Carlo reference $\widehat{V}^{\pi}(s_t)$ and evaluate the policy-consumed advantages of both methods against $\widehat{A}_t^{\pi}$.



(a) Absolute value error (b) Sign disagreement rate (c) Relative ranking

Figure 2: **Analysis of parametric step-level credit.** (a) Absolute value error of PPO's critic against $\widehat{V}^{\pi}(s_t)$, with the terminal-return baseline R. (b) Sign disagreement rate with $\widehat{A}_t^{\pi}$ for PPO and iStar. (c) Within-trajectory Spearman correlation between predicted and Monte-Carlo value movements. Bands denote 95% bootstrap confidence intervals.

Absolute value: state-specific predictions remain inaccurate. Figure 2a reports the mean absolute error of PPO's critic relative to $\widehat{V}^{\pi}(s_t)$. As a simple trajectory-level baseline, we assign the realized terminal return R to every state in its rollout. Despite producing state-specific predictions, the critic achieves comparable error to this trajectory-wide baseline. Accurate absolute value estimation is particularly challenging in long-horizon agentic tasks, where intermediate states receive no direct supervision and their eventual outcomes depend on many downstream decisions (Kazemnejad et al., 2025). Notably, the increase in MAE later in training reflects a broader target-value distribution as

the policy improves: the standard deviation of $\widehat{V}^{\pi}$ increases from 0.16 early in training to 0.33–0.52 at later checkpoints, expanding the range of values the critic must distinguish. The result therefore indicates that dense state-specific prediction alone does not ensure accurate absolute state values.

Direct credit: substantial disagreement in update direction. This error becomes consequential when learned predictions directly determine policy credit. Figure 2b shows that PPO's advantages disagree in sign with the Monte-Carlo reference for a substantial fraction of decisions throughout training. This pattern persists under both $\gamma = \lambda = 1$ and $\gamma = 0.99, \lambda = 0.95$ in GAE (Schulman et al., 2016). A similar rate of disagreement appears for iStar, whose learned step rewards are combined with GRPO-style outcome credit. Thus, both explicit critics and implicit reward models produce advantages that can reinforce actions with negative reference advantages or suppress those with positive ones.

Relative ranking: learned values retain useful within-trajectory information. We next examine whether PPO's critic preserves relative differences between actions. For an intermediate transition with no immediate reward, the one-step advantage is related to the value movement

$$\Delta V_t = \gamma V_\phi(s_{t+1}) - V_\phi(s_t). \tag{8}$$

We compare the within-trajectory ordering induced by ΔV_t with that of the Monte-Carlo movement $\Delta \widehat{V}^{\pi}_t = \gamma \widehat{V}^{\pi}(s_{t+1}) - \widehat{V}^{\pi}(s_t)$. Figure 2c shows consistently positive Spearman correlations across training checkpoints. Although modest, this agreement indicates that the critic retains useful information about the relative ordering of action quality within a trajectory, offering a signal for distinguishing more consequential actions from less consequential ones. Learned value movement can therefore serve as a useful signal for refining relative credit allocation, even when absolute value predictions and directly constructed advantages are unreliable. These findings motivate a constrained use of learned values: non-parametric credit remains the grounding signal, while learned value movement provides complementary information for refining its relative allocation.

4 GRAW: Grounded Relative-Value Advantage Weighting

Let $\mathcal{B}$ denote a non-parametric step-level base estimator and $A^{\mathcal{B}}_t$ the advantage it assigns to action a_t. Rather than constructing a new policy credit from the learned signal, GRAW applies a positive multiplicative reweighting to the base advantage:

$$A^{GRAW}_t = \alpha_t A^{\mathcal{B}}_t, \qquad \alpha_t > 0. \tag{9}$$

The positive multiplier α_t preserves each base advantage sign. We compute these multipliers from learned relative value movement and normalize them separately over positive- and negative-credit actions within each trajectory, preserving the total absolute credit in each direction. Learned predictions therefore adjust only the relative allocation of existing credit, without reversing its direction or supplying independent policy credit.

4.1 Learning a Relative Action Ranker

Because GRAW only requires relative action ranking, a separate critic would add substantial computation through an additional backbone. We therefore reuse the policy representation and train a lightweight value head on detached hidden states using outcome-derived targets.

Detached value head. Let h_t denote the last-layer hidden representation of state s_t immediately before generating action a_t. The head reuses this representation from the policy forward pass:

$$\bar{h}_t = \mathrm{sg}(h_t), \qquad V_\phi(s_t) = f_\phi(\bar{h}_t), \tag{10}$$

where f_ϕ is a two-layer MLP with a GELU activation (Hendrycks & Gimpel, 2016), and $\mathrm{sg}(\cdot)$ denotes stop-gradient. Here, $V_\phi(s_t) \in \mathbb{R}$ is the head's pre-sigmoid logit. Because the input is detached, the head loss updates only ϕ without directly modifying the policy representation.

Targets from multiple observed continuations. Using a single observed continuation as the training target for the value head can yield high variance, since its return also depends on the downstream decisions that happen to follow the current state. Rather than collecting additional continuations, we reuse the comparison groups $\mathcal{C}_k(s_{i,t})$ of the base estimator. A depth k is considered valid when

its comparison group contains visits from at least two distinct trajectories. For each valid depth, $\mu(\mathcal{C}_k(s_{i,t}))$ gives the mean realized return over the visits in that group. We combine these means using the same depth weights w_k as the base estimator, setting $w_k = 0$ for invalid depths and renormalizing over the valid ones. We define

$$y_{i,t} = \begin{cases} \sum_{k=0}^{K_{\mathrm{ctx}}} w_k \, \mu(\mathcal{C}_k(s_{i,t})), & \text{if any valid depth exists,} \quad t < T, \\ r_{i,t} + \gamma y_{i,t+1}, & \text{otherwise,} \quad t < T, \\ R^{(i)}, & t = T. \end{cases} \tag{11}$$

Thus, GiGPO uses only the anchor-state group ($K_{\mathrm{ctx}} = 0$, $w_0 = 1$), while HGPO uses its hierarchy weights $w_k \propto (k+1)^{\alpha}$ over the valid depths. When no valid comparison group is available, the successor target is propagated through the Bellman recursion (Bellman, 1966; Sutton et al., 1998). Every target therefore ultimately traces back to collected outcomes rather than bootstrapping from the head's own predictions. Since $y_{i,t} \in [0, 1]$, we apply a sigmoid to the head output when computing the training loss:

$$\mathcal{L}_{\mathrm{head}}(\phi) = \frac{1}{|\mathcal{D}|} \sum_{(i,t)\in\mathcal{D}} \mathrm{BCE}\big(\sigma(V_\phi(s_{i,t})), y_{i,t}\big), \tag{12}$$

where σ is the logistic sigmoid and $\mathcal{D}$ indexes state visits in the current rollout batch. We omit the trajectory index i when unambiguous hereafter.

Relative value movement. Under deterministic transitions, the true advantage at an intermediate step with $r_t = 0$ satisfies $A^\pi(s_t, a_t) = \gamma V^\pi(s_{t+1}) - V^\pi(s_t)$, relating action quality to value movement. Motivated by this relationship, we construct a relative action score from discounted movement in the head's logits:

$$m_t = \begin{cases} \gamma V_\phi(s_{t+1}) - V_\phi(s_t), & t < T, \\ (\gamma - 1) V_\phi(s_T), & t = T, \end{cases} \tag{13}$$

with the terminal boundary handled by setting $V_\phi(s_{T+1}) = V_\phi(s_T)$. This discounted logit movement serves only as a relative signal for redistributing existing credit within a trajectory, rather than as a direct estimate of the policy advantage.

4.2 Direction-Preserving Credit Redistribution

We redistribute credit only among actions that share the same base update direction. For a trajectory τ and each non-empty direction $d \in \{-1, +1\}$, we define

$$\mathcal{G}^d(\tau) = \big\{t \in \{1, \ldots, T_\tau\} : \mathrm{sign}(A_t^{\mathcal{B}}) = d\big\}, \qquad M^d(\tau) = \sum_{u\in\mathcal{G}^d(\tau)} |A_u^{\mathcal{B}}|. \tag{14}$$

Here, $\mathcal{G}^d(\tau)$ groups actions that the base estimator assigns the same update direction, while $M^d(\tau)$ denotes their total absolute credit. Actions with $A_t^{\mathcal{B}} = 0$ are left unchanged.

Direction-aware relative scoring. The same value movement should be interpreted differently depending on the grounded update direction. Among positively credited actions, those that move toward better states should be reinforced more; among negatively credited actions, those that move toward worse states should be suppressed more. We therefore orient the relative movement by the sign of the base advantage:

$$z_t = \frac{\mathrm{sign}(A_t^{\mathcal{B}})\, m_t}{\sigma_m}, \qquad \sigma_m = \max(\mathrm{std}_{u\in\mathcal{D}}(m_u), \epsilon), \tag{15}$$

where $\epsilon > 0$ is a small constant for numerical stability. Thus, a larger z_t indicates that action t should receive greater relative weight within its update-direction group.

Relative-value-guided redistribution. We use z_t to tilt the base credit allocation within each direction. For $t \in \mathcal{G}^d(\tau)$, we define

$$\alpha_t = \frac{M^d(\tau)\exp(z_t/\beta)}{\sum_{u\in\mathcal{G}^d(\tau)} |A_u^{\mathcal{B}}| \exp(z_u/\beta)}, \qquad A_t^{GRAW} = \alpha_t A_t^{\mathcal{B}}, \tag{16}$$

Table 1: **Main results on WebShop, SciWorld, and ALFWorld.** Results are reported as mean ± standard deviation over three random seeds using Qwen2.5-1.5B-Instruct.

Method	WebShop		SciWorld		ALFWorld		Average
	Score	Succ	Score	Succ	Seen	Unseen	
Prompting	22.16	6.25	5.72	1.56	9.29	5.97	8.49
PPO	83.91$_{\pm 2.67}$	66.67$_{\pm 1.19}$	35.73$_{\pm 10.33}$	27.08$_{\pm 8.64}$	65.71$_{\pm 10.38}$	57.46$_{\pm 11.73}$	56.09$_{\pm 3.88}$
GRPO	83.46$_{\pm 4.51}$	68.23$_{\pm 1.63}$	34.61$_{\pm 3.24}$	25.52$_{\pm 3.69}$	70.00$_{\pm 3.71}$	71.14$_{\pm 6.08}$	58.83$_{\pm 1.34}$
RLOO	82.38$_{\pm 4.13}$	63.80$_{\pm 2.39}$	44.41$_{\pm 2.65}$	35.16$_{\pm 2.82}$	68.10$_{\pm 8.58}$	64.18$_{\pm 9.35}$	59.67$_{\pm 3.47}$
iStar w/ GRPO	83.31$_{\pm 6.33}$	67.97$_{\pm 6.68}$	49.28$_{\pm 9.49}$	38.54$_{\pm 9.64}$	80.71$_{\pm 3.78}$	80.60$_{\pm 2.59}$	66.73$_{\pm 5.05}$
iStar w/ RLOO	82.67$_{\pm 1.78}$	67.45$_{\pm 1.97}$	38.30$_{\pm 6.42}$	27.60$_{\pm 6.31}$	84.52$_{\pm 1.49}$	81.59$_{\pm 1.55}$	63.69$_{\pm 1.26}$
GiGPO	85.79$_{\pm 1.66}$	71.35$_{\pm 4.30}$	52.86$_{\pm 3.56}$	43.23$_{\pm 3.16}$	88.81$_{\pm 2.18}$	84.83$_{\pm 1.14}$	71.15$_{\pm 1.41}$
+ GRAW	90.16$_{\pm 0.28}$	75.52$_{\pm 1.63}$	**55.44**$_{\pm 1.34}$	**48.18**$_{\pm 1.19}$	90.24$_{\pm 0.82}$	86.57$_{\pm 0.00}$	74.35$_{\pm 0.07}$
HGPO	88.13$_{\pm 0.68}$	74.48$_{\pm 1.80}$	52.87$_{\pm 3.22}$	45.57$_{\pm 5.54}$	91.19$_{\pm 2.51}$	88.81$_{\pm 2.59}$	73.51$_{\pm 1.83}$
+ GRAW	**90.83**$_{\pm 0.65}$	**77.08**$_{\pm 0.45}$	53.97$_{\pm 1.57}$	46.61$_{\pm 3.16}$	**91.90**$_{\pm 0.41}$	**89.30**$_{\pm 0.86}$	**74.95**$_{\pm 0.65}$

where $\beta > 0$ controls the strength of redistribution: larger values approach the base allocation, while smaller values place greater relative emphasis on actions with larger z_t. By construction, the redistribution preserves both the total absolute credit for each direction within a trajectory and the sign of every base advantage:

$$\sum_{t \in \mathcal{G}^d(\tau)} |A_t^{GRAW}| = M^d(\tau) = \sum_{t \in \mathcal{G}^d(\tau)} |A_t^{\mathcal{B}}|, \qquad \operatorname{sign}(A_t^{GRAW}) = \operatorname{sign}(A_t^{\mathcal{B}}). \tag{17}$$

Thus, learned relative values change only how existing credit is distributed among same-direction actions, without altering the direction or total absolute credit supplied by the base estimator.

Plug-in integration. GRAW modifies only the advantage passed to the base policy objective: $A_t^{\mathcal{B}}$ is replaced by A_t^{GRAW}, while rollout collection, base advantage estimation, policy loss, and the remaining optimization procedure are unchanged. The value-head loss updates only ϕ because its input representation is detached from the policy backbone. Thus, GRAW requires no additional rollouts or environment interactions and adds only a lightweight value head and advantage reweighting.

5 Experiments

5.1 Experimental Setup

We evaluate GRAW on three long-horizon agent benchmarks: WebShop (Yao et al., 2022), SciWorld (Wang et al., 2022), and ALFWorld (Shridhar et al., 2021), all of which provide policy supervision through terminal rewards. For WebShop and SciWorld, we report *score* and *success rate*; for ALFWorld, we report *success rate* on the seen and unseen splits. We compare against four groups of baselines: (i) prompting with the base model; (ii) trajectory-level group-relative RL, including GRPO (Shao et al., 2024) and RLOO (Ahmadian et al., 2024); (iii) parametric credit methods, including PPO (Schulman et al., 2017) with an explicit value critic and iStar (Liu et al., 2026) with GRPO or RLOO using implicit step rewards; and (iv) non-parametric step-level methods, including GiGPO (Feng et al., 2025) and HGPO (He et al., 2026a). We use Qwen2.5-1.5B-Instruct (Yang et al., 2024) for the main comparison and further evaluate Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct to examine whether the gains persist across model scales. All methods are trained for 150 policy-update steps using sampling temperatures of 1.0 for training and 0.4 for evaluation. GRAW is applied to GiGPO and HGPO as a plug-in, adding only the lightweight relative-value head and credit redistribution while leaving the underlying rollout and policy-optimization procedures unchanged. Detailed experimental settings are provided in Appendix B.

5.2 Effectiveness of Direction-Preserving Redistribution

Table 1 shows that GRAW consistently improves strong non-parametric step-level baselines across all three benchmarks and every reported metric. Prompting alone performs poorly, while trajectory-level group-relative methods such as GRPO and RLOO provide substantial gains. Among parametric step-level methods, PPO exhibits relatively large variance across seeds, reflecting the difficulty of using learned values directly as policy credit. iStar avoids explicit value estimation through implicit

step rewards and improves over its GRPO and RLOO bases, but its gains remain limited because the learned rewards still directly modify policy credit. The non-parametric step-level methods GiGPO and HGPO achieve stronger overall results than these evaluated parametric methods. With GRAW, average performance increases from 71.15 to 74.35 for GiGPO (+3.20) and from 73.51 to 74.95 for HGPO (+1.44), with the latter achieving the best overall result among the evaluated methods. GRAW also reduces variance across seeds for every reported metric relative to its corresponding base estimator. The larger gain on GiGPO is consistent with its broader redistributable credit: unlike HGPO, which relies solely on hierarchical step-level contrasts and leaves roughly 20% of actions without eligible credit, GiGPO retains a trajectory-level component shared across the episode. This allows GRAW to redistribute credit across more actions in GiGPO, while the gains on HGPO show that it remains effective even when fewer actions are eligible for redistribution. Training dynamics in Appendix C.1 further show that these improvements persist throughout training.

Computational Efficiency. GRAW reuses policy hidden states through a lightweight value head and therefore requires neither an additional backbone forward pass nor extra rollouts. On ALFWorld, which has the longest trajectories, the added value-head and redistribution computation remains minor relative to the base training cost; a detailed wall-clock breakdown is provided in Appendix C.2.

5.3 Generalization Across Model Scales

Table 2 examines whether the gains from GRAW persist as the policy scales from Qwen2.5-3B to 7B. For compact comparison, we report score on WebShop and SciWorld and seen success rate on ALFWorld; full results are provided in Appendix C.3. GRAW improves GiGPO and HGPO by 2.83 and 3.29 points on 3B, with larger gains of 4.11 and 5.05 points on 7B, respectively. These results show that relative-value-guided redistribution remains effective across model scales, with its gains preserved and even amplified at 7B.

Table 2: **Performance across model scales.**

Method	WebShop	SciWorld	ALFWorld	Average
		Qwen2.5-3B-Instruct		
GiGPO	87.66	50.37	88.57	75.53
+ GRAW	88.85$_{(+1.19)}$	55.53$_{(+5.16)}$	90.71$_{(+2.14)}$	78.36$_{(+2.83)}$
HGPO	87.24	54.56	85.71	75.84
+ GRAW	90.00$_{(+2.76)}$	57.38$_{(+2.82)}$	90.00$_{(+4.29)}$	79.13$_{(+3.29)}$
		Qwen2.5-7B-Instruct		
GiGPO	88.66	51.90	90.00	76.85
+ GRAW	91.21$_{(+2.55)}$	56.68$_{(+4.78)}$	95.00$_{(+5.00)}$	80.96$_{(+4.11)}$
HGPO	89.62	53.63	93.57	78.94
+ GRAW	92.18$_{(+2.56)}$	63.37$_{(+9.74)}$	96.43$_{(+2.86)}$	83.99$_{(+5.05)}$

5.4 Alignment with Relative Action Contribution

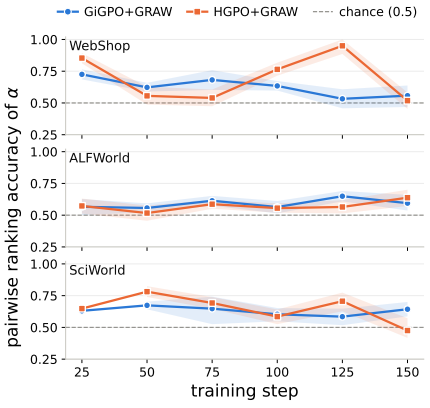


Figure 3: **Alignment of learned multipliers with relative action contribution.** Pairwise ranking accuracy of α_t against environment-derived progress signals; bands denote 95% bootstrap confidence intervals.

Task-level gains alone do not show whether GRAW redistributes credit toward the actions that matter most within a trajectory. Using Qwen2.5-1.5B-Instruct, we therefore evaluate the ordering induced by α_t against environment-derived progress signals. Within each same-direction group $\mathcal{G}^d(\tau)$, we compare all action pairs. If action i contributes more than action j, a correct redistribution should assign $\alpha_i > \alpha_j$ in a positive-credit group; in a negative-credit group, the action causing greater harm should receive the larger multiplier. Pairwise ranking accuracy measures the fraction of correctly ordered pairs, with 0.5 corresponding to chance. We evaluate checkpoints from 25 to 150 every 25 steps; Appendix B.5 provides detailed metric definitions and progress signals.

Figure 3 shows that α_t consistently captures relative action contribution across environments and base estimators. Averaged over six checkpoints, pairwise ranking accuracy reaches 0.63/0.70 on WebShop, 0.59/0.57 on ALFWorld, and 0.63/0.65 on SciWorld for GiGPO/HGPO, respectively. Since GRAW uses learned values only for within-direction reweighting, this moderate signal is sufficient to refine grounded credit: an overall accuracy of about 0.63 reduces pairwise ranking error by roughly 26% relative to random ordering, and 35 of

36 settings exceed chance. Qualitative examples illustrating how α_t ranks actions within trajectories are provided in Appendix C.4.

5.5 Ablating the Sources of Improvement

We ablate three design choices of GRAW on GiGPO using Qwen2.5-1.5B-Instruct with a single seed. *w/o redistribution* (Eq. 16) replaces direction-preserving reweighting with iStar-style additive integration of the learned value movement m_t into the base credit. *w/o group targets* (Eq. 11) trains the value head from a single realized continuation, analogous to conventional PPO critic training. *w/o representation reuse* (Eq. 10) replaces the lightweight shared head with a separate critic backbone.

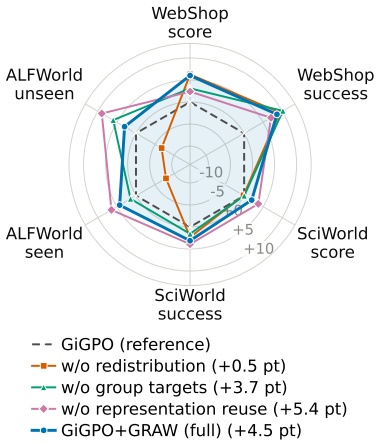


Figure 4: **Ablation of GRAW on GiGPO.**

The full method improves GiGPO by $+4.5$ points on average, while removing direction-preserving redistribution reduces the gain to $+0.5$ points. Additive integration can reverse the base advantage sign; this effect grows with trajectory length, peaking on ALFWorld, and affects roughly 10% of actions on average. Removing grouped targets still yields a $+3.7$-point gain, showing that relative-value-guided redistribution itself remains effective even with single-continuation supervision. A separate critic achieves the largest gain ($+5.4$ points) but requires an additional backbone forward and backward pass, revealing an accuracy–computation trade-off. HGPO shows the same trends (Appendix C.5).

6 Related Work

Group-relative and step-level RL for LLM agents. Group-relative methods construct policy advantages from outcome comparisons across sampled rollouts (Shao et al., 2024; Ahmadian et al., 2024). Recent work extends such credit to finer granularity through step-level sampling (Dong et al., 2026), tree-structured grouping (Ji et al., 2026), selective step-level optimization (Qi et al., 2026), hindsight refinement (Tan et al., 2026), hierarchical planning–execution credit (Peng et al., 2026), graph-based aggregation (Wang et al., 2026), and adaptive granularity (Liang et al., 2026). Within this line, GiGPO (Feng et al., 2025) and HGPO (He et al., 2026a) construct explicit non-parametric step-level advantages from sampled experience. GRAW is complementary to these methods: we retain this grounded credit and use learned relative value movement only to reweight its allocation.

Parametric credit assignment for LLM agents. PPO (Schulman et al., 2017) constructs advantages from learned values, while process reward models score intermediate reasoning steps or agent actions to guide selection or optimization (Wang et al., 2024; Lightman et al., 2024; Li & Li, 2025; Xi et al., 2026). Recent methods also derive process signals from outcome supervision or incorporate learned step rewards directly into policy optimization (Yuan et al., 2025; Cui et al., 2025; Liu et al., 2026; Xu et al., 2025). Although these methods differ in how their learned signals are obtained, the resulting predictions are generally used to define or modify the policy credit itself. In contrast, GRAW does not use learned predictions as independent policy credit; it only reweights existing non-parametric credit while preserving its signs and total absolute credit in each direction.

7 Conclusion

We identify a complementary asymmetry in step-level credit: non-parametric advantages provide informative update directions but imperfect allocation, whereas learned value movement retains useful relative information despite being unreliable as direct policy credit. Based on this observation, we introduce GRAW, which uses learned relative values only for within-direction reweighting while preserving every base advantage sign and the total absolute credit in each direction. Across three long-horizon agent benchmarks and multiple model scales, GRAW consistently improves strong step-level baselines with lower variance across seeds and minor computational overhead, showing that each credit signal can be restricted to the role it supports most reliably.

AI USE STATEMENT

In this work, we used generative AI tools to aid or polish writing, for retrieval and discovery, and to draft sections of the paper. We have not used generative AI tools for research ideation or execution, for generating synthetic datasets, and for proving mathematical claims. We have reviewed all AI-assisted work. All AI-assisted text was carefully checked and revised by the authors, and literature retrieved with AI assistance was verified against the original sources. Technical claims, equations, experimental results, and citations were independently checked by the authors for correctness. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed information to support reproducibility throughout the main paper and appendix. The complete formulation of the proposed method and its integration with the base estimators are described in Section 4, with the full training procedure summarized in Algorithm 1. The appendix provides detailed formulations of the credit estimators, benchmark and evaluation protocols, baseline configurations, training hyperparameters, and implementation details of GRAW. Upon acceptance, we will publicly release the source code, training and evaluation scripts, and configuration files required to reproduce the reported results.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Richard Bellman. Dynamic programming. *science*, 153(3731):34–37, 1966.

Jae-Woo Choi, Youngwoo Yoon, Hyobin Ong, Jaehong Kim, and Minsu Jang. Lota-bench: Benchmarking language-oriented task planners for embodied agents. In *International Conference on Learning Representations*, volume 2024, pp. 52718–52744, 2024.

Marc-Alexandre Côté, Akos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. Textworld: A learning environment for text-based games. In *Workshop on computer games*, pp. 41–75. Springer, 2018.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. In *International Conference on Learning Representations*, volume 2026, pp. 16981–17017, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *Advances in Neural Information Processing Systems*, 38:46375–46408, 2025.

Apurva Gandhi and Graham Neubig. Go-browse: Training web agents with structured exploration. In *International Conference on Learning Representations*, volume 2026, pp. 138973–138992, 2026.

Shuo He, Lang Feng, Xin Cheng, Lei Feng, Bo An, et al. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. In *International Conference on Learning Representations*, volume 2026, pp. 27572–27593, 2026a.

Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Qi Gu, Hui Su, and Xunliang Cai. Vitabench: Benchmarking llm agents with versatile interactive tasks in real-world applications. In *International Conference on Learning Representations*, volume 2026, pp. 91654–91681, 2026b.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Ethan Hsu, Hong Meng Yam, Ines Bouissou, Aaron John, Raj Thota, Josh Koe, Vivek Putta, G Dharesan, Alexander Spangher, Shikhar Murty, et al. Webds: An end-to-end benchmark for web-based data science. In *International Conference on Learning Representations*, volume 2026, pp. 31777–31795, 2026.

Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. Tree search for llm agent reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 87362–87388, 2026.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms. 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Wendi Li and Yixuan Li. Process reward model with q-value rankings. In *International Conference on Learning Representations*, volume 2025, pp. 14708–14726, 2025.

Taoran Liang, Yang Liu, Shang Luo, Yingguang Yang, Rongrong Zhang, Yingzong Min, Yulin Huang, Jianshen Zhang, Yongzhi Qi, Kefu Xu, et al. Granularity-adaptive credit assignment for long-horizon llm agent reinforcement learning. *arXiv preprint arXiv:2609.12424*, 2026.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Xiaoqian Liu, Ke Wang, Yuchuan Wu, Fei Huang, Yongbin Li, Jianbin Jiao, and Junge Zhang. Agentic reinforcement learning with implicit step rewards. In *International Conference on Learning Representations*, volume 2026, pp. 129271–129291, 2026.

Priyam Parashar, Jacob Krantz, Matthew Chang, Kavit Shah, Xavier Puig, and Roozbeh Mottaghi. Planning with an embodied learnable memory. In *International Conference on Learning Representations*, volume 2026, pp. 142387–142400, 2026.

Jiangweizhi Peng, Yuanxin Liu, Ruida Zhou, Charles Fleming, Zhaoran Wang, Alfredo Garcia, and Mingyi Hong. Hiper: Hierarchical reinforcement learning with explicit credit assignment for large language model agents. *International Conference on Machine Learning*, 2026.

Qiuyi Qi, Tian Liang, Mutian Bao, Jinjian Zhang, Dongnan Liu, Wei Zhou, Linjian Mo, Ming Kong, Jie Liu, Feng Zhang, et al. Stapo: Selective trajectory-aware policy optimization for llm agent training. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28371–28392, 2026.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *International Conference on Learning Representations*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Jiaqi Shao, Yuxiang Lin, Munish P Lohani, Yufeng Miao, and Bing Luo. Do llm agents know how to ground, recover, and assess? evaluating epistemic competence in information-seeking agents. In *International Conference on Learning Representations*, volume 2026, pp. 152647–152679, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10737–10746. IEEE, 2020.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.03768.

Richard S Sutton, Andrew G Barto, and Andrew Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon llm agents. *arXiv preprint arXiv:2603.08754*, 2026.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your agent smarter than a 5th grader? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.775. URL https://aclanthology.org/2022.emnlp-main.775/.

Yunan Wang, Minghui Song, Zihan Zhang, Shaohan Huang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. Group-graph policy optimization for long-horizon agentic reinforcement learning. *arXiv preprint arXiv:2606.22995*, 2026.

Zhiheng Xi, Chenyang Liao, Guanyu Li, Zhihao Zhang, Wenxiang Chen, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, et al. Agentprm: Process reward models for llm agents via step-wise promise and progress. In *Proceedings of the ACM Web Conference 2026*, pp. 4184–4195, 2026.

Weimin Xiong, Yifan Song, Xiutian Zhao, Wenhao Wu, Xun Wang, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. Watch every step! llm agent learning via iterative step-level process refinement. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1556–1572, 2024.

Huimin Xu, Xinnian Mao, Feng-Lin Li, Xiaobao Wu, Wang Chen, Wei Zhang, and Luu Anh Tuan. Full-step-dpo: Self-supervised preference optimization with step-wise rewards for mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24343–24356, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. In *International Conference on Machine Learning*, 2025.

Chenchen Zhang. From reasoning to agentic: Credit assignment in reinforcement learning for large language models. *arXiv preprint arXiv:2604.09459*, 2026.

Appendix

A Detailed Formulation of Step-Level Credit Estimators

A.1 Agentic Reinforcement Learning

We consider an LLM agent that interacts with an environment to complete a multi-step task $x \sim p(\mathcal{X})$. At each step $t = 1, \ldots, T$, the agent observes a state $s_t \in \mathcal{S}$ and generates a textual action $a_t \in \mathcal{V}^{\leq n}$ according to $\pi_\theta(a_t \mid s_t, x)$, where $\mathcal{V}$ denotes the token vocabulary and n is the maximum generation length. The environment returns a scalar reward $r_t \in \mathbb{R}$ and transitions to s_{t+1}. A complete interaction forms a trajectory $\tau = ((s_1, a_1, r_1), \ldots, (s_T, a_T, r_T))$.

We focus on sparse terminal supervision, where $r_t = 0$ for $t < T$ and $r_T = R \in [0, 1]$. The realized return-to-go is

$$G_t = \sum_{k=t}^{T} \gamma^{k-t} r_k = \gamma^{T-t} R, \tag{18}$$

where $\gamma \in (0, 1]$ is the discount factor. For a fixed task x, the action-value function $Q^\pi(s_t, a_t)$ measures the expected return after taking a_t at s_t and following the current policy thereafter, while the state-value function $V^\pi(s_t)$ additionally averages over the action sampled at s_t (Sutton et al., 1998):

$$\begin{aligned} Q^\pi(s_t, a_t) &= \mathbb{E}[G_t \mid s_t, a_t, x], \\ V^\pi(s_t) &= \mathbb{E}_{a \sim \pi(\cdot \mid s_t, x)}[Q^\pi(s_t, a)]. \end{aligned} \tag{19}$$

We suppress task dependence in the value-function notation. The corresponding population advantage is

$$A^\pi(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t), \tag{20}$$

which measures the expected benefit or disadvantage of taking a_t relative to actions drawn from the current policy at the same state.

In practice, RL algorithms construct an estimator-specific advantage A_t to approximate A^π; we refer to A_t as *policy credit*. Omitting algorithm-specific importance weighting, clipping, and regularization, its contribution to the policy update can be written schematically as

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=1}^{T} A_t \nabla_\theta \log \pi_\theta(a_t \mid s_t, x) \right]. \tag{21}$$

The sign of A_t determines whether the sampled action is reinforced or suppressed, while $|A_t|$ weights its gradient contribution. Throughout the paper, *update direction* refers to this action-level sign, while *credit allocation* refers to how credit magnitudes are distributed across actions.

A.2 Non-Parametric Step-Level Credit

For each task x, group-relative RL samples N trajectories $\{\tau^{(i)}\}_{i=1}^{N}$ from the current policy and constructs policy credit directly from comparisons of their realized outcomes (Shao et al., 2024; Ahmadian et al., 2024). Non-parametric step-level methods such as GiGPO (Feng et al., 2025) and HGPO (He et al., 2026a) extend these comparisons to actions taken from the same state under different amounts of shared history.

We denote a non-parametric step-level base estimator by $\mathcal{B}$ and write its final credit for step t of trajectory i as

$$A^{\mathcal{B}}_{i,t} = \lambda_E A^E_i + \sum_{k=0}^{K_{\text{ctx}}} w_k \, A^{S,k}_{i,t}, \tag{22}$$

where A^E_i is the trajectory-level empirical credit shared by all actions of $\tau^{(i)}$, $A^{S,k}_{i,t}$ is the step-level empirical credit at context depth k, w_k weights that depth, and K_{ctx} is the maximum context depth considered by the estimator.

Trajectory-level credit. Trajectory-level credit compares the realized terminal outcome of trajectory i with outcomes from the same rollout group. Its generic form is

$$A_i^E = \frac{R^{(i)} - b_i^E}{F_{\text{norm}}^E}, \tag{23}$$

where b_i^E is the group-relative baseline and F_{norm}^E is the corresponding normalization factor. For example, GRPO uses the group mean as the baseline and the group standard deviation as normalization, whereas RLOO uses the mean outcome of the remaining trajectories as a leave-one-out baseline without variance normalization. In either case, A_i^E is shared by all actions of trajectory $\tau^{(i)}$.

Step-level credit. At context depth k, let $\mathcal{C}_k(s_{i,t})$ denote the sampled state–action pairs in the rollout group whose current state and k preceding states coincide with $(s_{i,t-k}, \ldots, s_{i,t})$. The step-level credit compares the realized return following $a_{i,t}$ against returns observed in this comparison group:

$$A_{i,t}^{S,k} = \frac{G_{i,t} - \mu\big(\mathcal{C}_k(s_{i,t})\big)}{F_{\text{norm}}\big(\mathcal{C}_k(s_{i,t})\big)}, \tag{24}$$

where $\mu(\mathcal{C}_k(s_{i,t}))$ denotes the mean realized return over the state–action pairs in $\mathcal{C}_k(s_{i,t})$, and $F_{\text{norm}}(\mathcal{C}_k(s_{i,t}))$ denotes the estimator-specific normalization factor computed over the same returns, either their standard deviation or 1.

GiGPO uses both trajectory- and step-level credit, with $\lambda_E = 1$, and considers only the anchor-state group ($K_{\text{ctx}} = 0$, $w_0 = 1$). HGPO removes the trajectory-level component by setting $\lambda_E = 0$ and instead aggregates step-level contrasts across multiple context depths. Specifically, its weights satisfy

$$w_k \propto (k+1)^\alpha, \tag{25}$$

with normalization over the context depths for which a valid comparison group exists. For notational simplicity, we suppress the dependence of the normalized weights on the particular step (i, t).

The distinction from the population action value is important. Although the comparison baseline $\mu(\mathcal{C}_k(s_{i,t}))$ pools returns from multiple sampled occurrences, the action-specific quantity $G_{i,t}$ is obtained from the single continuation actually observed after $a_{i,t}$. In contrast,

$$Q^\pi(s_{i,t}, a_{i,t}) = \mathbb{E}[G_{i,t} \mid s_{i,t}, a_{i,t}, x] \tag{26}$$

averages over possible future continuations under the current policy. The empirical return $G_{i,t}$ therefore reflects not only the current action but also the downstream decisions that happen to follow it. This dependence motivates the analysis in Section 3.1. We omit the trajectory index i when it is unambiguous hereafter.

A.3 Parametric Step-Level Credit

Parametric methods instead learn state- or action-specific predictions from accumulated experience and use these predictions to construct policy credit. We consider two representative forms: explicit value critics and implicit step-reward models. In both cases, a learned predictor directly enters the construction of the advantage consumed by the policy objective.

Explicit value critic. PPO (Schulman et al., 2017) uses a learned state-value function $V_\phi(s_t)$ to approximate $V^\pi(s_t)$ and constructs advantages through generalized advantage estimation (GAE) (Schulman et al., 2016). The temporal-difference residual is

$$\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t), \tag{27}$$

and the GAE advantage is

$$A_t^{\text{PPO}} = \sum_{\ell=0}^{T-t} (\gamma\lambda)^\ell \delta_{t+\ell}, \tag{28}$$

where $\lambda \in [0, 1]$ controls the bias–variance trade-off. The critic provides a state-specific prediction at each visited state, and these predictions enter the policy advantage through the temporal-difference residuals. Errors in the learned values can therefore affect both the sign and magnitude of the resulting policy credit.

Implicit step reward. Implicit reward methods provide an alternative to explicit state-value prediction by learning intermediate reward signals from trajectory-level supervision. Methods such as PRIME (Cui et al., 2025) and iStar (Liu et al., 2026) incorporate these learned signals into policy advantages without requiring an explicit state-value critic.

For example, iStar ranks sampled trajectories by their outcomes to construct preference pairs (τ^+, τ^-) and trains an implicit process-reward model with a trajectory-level DPO objective:

$$\mathcal{L}_{\mathrm{DPO}} = -\log \sigma\left(\beta_{\mathrm{DPO}} \log \frac{\pi_\psi(\tau^+ \mid x)}{\pi_{\theta_{\mathrm{old}}}(\tau^+ \mid x)} - \beta_{\mathrm{DPO}} \log \frac{\pi_\psi(\tau^- \mid x)}{\pi_{\theta_{\mathrm{old}}}(\tau^- \mid x)}\right), \tag{29}$$

where π_ψ denotes the implicit process-reward model, $\pi_{\theta_{\mathrm{old}}}$ is the previous policy snapshot, σ is the logistic sigmoid, and $\beta_{\mathrm{DPO}} > 0$ is a scaling coefficient. The trajectory log-ratio is understood as the sum of action log-ratios along the recorded trajectory. The corresponding implicit reward for action $a_{i,t}$ is

$$r^{\mathrm{imp}}_{i,t} = \beta_{\mathrm{DPO}} \log \frac{\pi_\psi(a_{i,t} \mid s_{i,t}, x)}{\pi_{\theta_{\mathrm{old}}}(a_{i,t} \mid s_{i,t}, x)}, \tag{30}$$

which is converted into a step-level advantage $A^{\mathrm{imp}}_{i,t}$. iStar combines this learned component with trajectory-level outcome credit:

$$A^{\mathrm{iStar}}_{i,t} = A^E_i + \lambda_{\mathrm{imp}} A^{\mathrm{imp}}_{i,t}, \tag{31}$$

where λ_{imp} controls the contribution of the learned step-level credit. The precise reward-learning objective and the conversion from implicit rewards to policy advantages are method-specific; their common feature is that learned action-specific predictions directly shape the policy credit.

Explicit critics and implicit reward models therefore construct step-specific credit through different routes: the former predicts state values and derives advantages through temporal-difference residuals, whereas the latter learns intermediate reward signals from trajectory outcomes. Section 3.2 examines which aspects of these learned credit signals can be trusted for step-level credit assignment.

B Experimental Setup Details

This appendix provides detailed experimental settings, including benchmark descriptions (Section B.1), evaluation metrics (Section B.2), baseline details (Section B.3), and implementation details (Section B.4).

B.1 Benchmarks

We evaluate GRAW on three representative long-horizon LLM-agent benchmarks: WebShop (Yao et al., 2022), SciWorld (Wang et al., 2022), and ALFWorld (Shridhar et al., 2021).

WebShop. WebShop (Yao et al., 2022) evaluates language agents in a simulated online shopping environment. Given a natural-language shopping request, the agent must search for relevant products, inspect candidate items, compare their attributes and options, and make a final purchase decision. The environment is built from a large-scale catalog of 1.18 million real-world products and more than 12,000 crowd-sourced user instructions, requiring agents to combine information retrieval, web navigation, and multi-step decision making. At each turn, the agent interacts with the website through textual actions, such as issuing a search query or clicking an available button. These actions move the agent across different page types, including search result pages and product detail pages. After the agent purchases an item or the episode terminates, WebShop provides a score measuring how well the selected product matches the user instruction. For policy optimization, we use a binary terminal reward indicating success or failure, while reporting the original score as an evaluation metric.

SciWorld. SciWorld (Wang et al., 2022) evaluates language agents in a text-based interactive environment designed around elementary science tasks. Given a natural-language goal, the agent must navigate through different rooms, inspect and manipulate objects, and perform a sequence of actions that satisfy the specified scientific objective. The benchmark covers a broad range of domains, including matter, electricity, forces, biology, and thermodynamics, requiring agents to combine commonsense reasoning, scientific knowledge, and long-horizon planning. At each turn, the agent

receives a textual observation describing its current surroundings and interacts with the environment through admissible textual actions, such as moving between locations, examining objects, or applying tools and operations. Task completion often requires multiple dependent steps, where earlier actions establish the conditions necessary for later ones. The environment provides task-specific scores based on progress toward the goal. For policy optimization, we use a binary terminal reward indicating task success or failure, while retaining the continuous final score for evaluation.

ALFWorld. ALFWorld (Shridhar et al., 2021) evaluates language agents on long-horizon household tasks in a text-based environment grounded in the embodied ALFRED benchmark (Shridhar et al., 2020). The benchmark includes diverse interaction patterns such as picking up and placing objects, cleaning, heating, cooling, and examining items, requiring agents to reason over object states, spatial relations, and multi-step dependencies. Each task is specified by a natural-language instruction, and the agent must complete it by issuing a sequence of high-level actions, such as `goto`, `take`, `open`, or `put`. Successful completion typically requires executing several actions in the correct order while maintaining consistency with the task goal. We use the final task-completion outcome as the terminal reward for policy optimization.

B.2 Evaluation Metrics

We adopt the standard evaluation metrics established for each benchmark to ensure direct comparability with prior work.

WebShop and SciWorld. For both benchmarks, we report two complementary metrics. *Score* is the mean terminal score $R \in [0, 1]$ across all evaluation tasks, capturing the degree of partial task completion (e.g., how closely the purchased product matches the user request in WebShop or how much progress the agent makes toward the scientific task goal in SciWorld). *Success rate* is the fraction of episodes that achieve the maximum reward $R = 1$, capturing strict task completion. Reporting both metrics distinguishes methods that improve partial-credit performance from those that improve outright success.

ALFWorld. Following the standard ALFWorld evaluation protocol, we report *success rate* on two splits separately. The *seen* split (140 episodes) measures in-domain generalization to familiar environment layouts encountered during training, while the *unseen* split (134 episodes) measures out-of-domain generalization to held-out environments and task instances. Reporting both splits allows us to disentangle in-distribution improvement from generalization to novel environments.

B.3 Baseline Details

All baselines are re-implemented in a unified `verl-agent` (Feng et al., 2025) codebase and evaluated under the same training and evaluation pipeline, so performance differences primarily reflect methodological choices rather than implementation artifacts.

Prompting. We evaluate the instruction-tuned policy without gradient updates, using the same prompts, action parser, interaction horizon, and evaluation set as the trained methods. This corresponds to the step-0 evaluation of each run and ensures that all methods start from the same policy.

GRPO (Shao et al., 2024). GRPO assigns a trajectory-level advantage by centering each rollout return against the group mean and normalizing by the group standard deviation. The resulting advantage is shared across all actions in the trajectory and therefore provides no step-specific credit. We use the reference implementation without algorithmic modification.

RLOO (Ahmadian et al., 2024). RLOO similarly constructs trajectory-level credit, but uses a leave-one-out baseline: each trajectory return is compared against the mean return of the remaining rollouts in its group, without variance normalization. We use the reference implementation without algorithmic modification.

PPO (Schulman et al., 2017). PPO obtains step-level credit from an explicit value critic, implemented as a separate network initialized from the same backbone and jointly trained with the policy. Policy advantages are computed using generalized advantage estimation. Following the reference configuration, PPO uses the same number of trajectories per update as the group-based methods, but samples

one trajectory per task because its baseline is provided by the learned critic rather than within-task rollout comparisons. PPO serves as our explicit parametric-credit baseline.

iStar (Liu et al., 2026). iStar learns an implicit process-reward model from preference pairs constructed within each rollout group and converts the learned scores into step-level advantages, which are combined with the trajectory-level advantage of the underlying base estimator. We evaluate iStar with both GRPO and RLOO and follow the authors' released hyperparameters. Each configuration is derived from the corresponding base-method script by adding only the iStar-specific settings, so that differences from the base estimator isolate the effect of implicit step-level credit.

GiGPO (Feng et al., 2025). GiGPO constructs non-parametric step-level credit by grouping actions that share the same anchor state and contrasting their realized discounted returns. This step-level signal is combined with a trajectory-level advantage shared across the episode. We follow the reference configuration, including the step-level discount, unit step-advantage weight, and environment-specific group normalization; for environments not specified in the reference implementation, we use its WebShop configuration.

HGPO (He et al., 2026a). HGPO extends non-parametric step-level credit to a hierarchy of groups defined by recent state history and removes the trajectory-level component, so its advantage is determined entirely by hierarchical step contrasts. We use the authors' default settings for the evaluated environments. To ensure a controlled comparison with GiGPO, we derive the HGPO configuration from the GiGPO script by changing only the estimator and its algorithm-specific settings.

B.4 Implementation Details

B.4.1 Shared Training Configuration

Unless otherwise noted, all methods including GRAW follow the common training and evaluation configuration summarized in Table 3, with method-specific differences restricted to rollout grouping and credit assignment. Rollouts are generated with vLLM (Kwon et al., 2023) under the current policy. Each update uses 128 trajectories. Group-based methods sample 16 tasks with 8 rollouts per task, whereas PPO samples 128 distinct tasks with one rollout each because its learned critic provides the baseline without requiring within-task rollout comparisons. Credit-assignment settings are inherited from the corresponding base method, and the additional settings specific to GRAW are described in Section B.4.2.

B.4.2 GRAW Implementation

Value head. The value head in Eq. 10 is a two-layer MLP that maps the backbone hidden size H to $H/4$, applies GELU, and projects to a scalar score $V_\phi(s_t)$. A sigmoid is applied only when computing the value-head loss in Eq. 12. The output layer is initialized near zero, yielding $\sigma(V_\phi) \approx 0.5$ at the start of training and therefore an initially weak relative-value signal. The head takes the hidden state at the final prompt token, i.e., the representation of state s_t immediately before action generation. This representation is obtained from the log-probability forward pass already required for the policy update, so GRAW introduces no additional backbone forward pass. The representation is detached before entering the value head, ensuring that $\mathcal{L}_{\text{head}}$ updates only ϕ and does not modify the policy representation. The head is optimized with the same optimizer as the policy but in a separate parameter group with a learning rate of 10^{-4}. Across the backbones used in our experiments ($H = 1536$–3584), the head adds only 0.6M–3.2M parameters, corresponding to less than 0.05% of the backbone.

Targets and relative value movement. Targets are constructed according to Eq. 11 on the normalized outcome scale $R \in [0, 1]$. Repeated states are aggregated within rollouts of the same task according to the grouping structure of the underlying base estimator. For GiGPO, we pool repeated visits to the same anchor state. For HGPO, pooling is performed at each hierarchy level, and levels supported by at least two distinct trajectories are combined using the same weighting scheme as HGPO's step-level contrasts. Relative value movement in Eq. 13 is computed from the pre-sigmoid scores V_ϕ and normalized by its standard deviation over the current rollout batch, as defined in Eq. 15.

Redistribution schedule. The temperature β in Eq. 16 controls the strength of relative-value-guided redistribution. During training, we linearly decay the inverse temperature $1/\beta$, applying stronger

Table 3: **Training and evaluation configuration.** Entries separated by "/" differ across WebShop / SciWorld / ALFWorld, respectively. Unless otherwise noted, settings are shared across methods and random seeds.

Setting	Value
Data and rollouts	
Training tasks per update	16
Rollout group size	8
Trajectories per update	128
Evaluation tasks	128 / 128 / 140 seen + 134 unseen
Evaluation interval	5
Environment	
Maximum steps per episode	15 / 50 / 50
Observation history length	2
Outcome reward R	Binary (success / failure)
Invalid-action penalty	1% of maximum return
Credit assignment	
Step-level discount γ	0.95
Step-advantage weight	1.0
Maximum context depth K_{ctx}	2
Optimization	
Policy learning rate	1×10^{-6}
Optimizer	AdamW, constant schedule
Weight decay	0.01
KL loss coefficient	0.01
Mini-batch size	64 / 256 / 256
Micro-batch size per GPU	8 / 32 / 32
Policy updates	150
Sequences and sampling	
Maximum prompt length	4096 / 8192 / 2048
Maximum response length	512
Training temperature	1.0
Evaluation temperature	0.4
Hardware	
GPUs per run	2× H200 (1.5B, 3B) / 4× H200 (7B)

redistribution early in training, when diverse rollouts provide richer ranking signals, and gradually recovering the base allocation as the policy becomes more concentrated (e.g., $\beta = 10 \to 30$). As optimization progresses, the policy increasingly concentrates probability mass on a narrower set of high-reward actions, producing more similar rollouts and reducing variation in observed continuations. This annealing therefore reduces the risk of over-amplifying small or noisy relative-value differences later in training, when the ranking signal becomes less discriminative. The schedule preserves the sign of the underlying advantage, and all other training and optimization settings are inherited from the corresponding base estimator.

B.5 Credit Redistribution Quality: Evaluation Protocol

This section details the pairwise ranking accuracy used in Section 5.4 and the environment-derived progress signals used to evaluate action contribution.

Evaluated trajectories. For each checkpoint $k \in \{25, 50, \ldots, 150\}$ of GiGPO+GRAW and HGPO+GRAW with Qwen2.5-1.5B (seed 0), we use the training batch collected by the policy at step k. Each batch contains 16 tasks with 8 rollouts per task, yielding 128 trajectories.

Pairwise ranking accuracy. Let Δ_t denote the environment-derived progress change induced by action a_t, as defined separately for each environment below. For each non-empty same-direction group $\mathcal{G}^d(\tau)$ in Eq. 14, we form all pairs of actions with distinct progress,

$$\mathcal{P}^d(\tau) = \big\{(i,j) : i,j \in \mathcal{G}^d(\tau),\ i < j,\ \Delta_i \neq \Delta_j\big\}, \tag{32}$$

and compute

$$\mathrm{Acc}^d(\tau) = \frac{1}{|\mathcal{P}^d(\tau)|} \sum_{(i,j)\in\mathcal{P}^d(\tau)} \Big[\mathbf{1}\big\{d(\Delta_i - \Delta_j)(\alpha_i - \alpha_j) > 0\big\} + \tfrac{1}{2}\mathbf{1}\big\{\alpha_i = \alpha_j\big\}\Big]. \tag{33}$$

For positive credit ($d = +1$), a pair is ranked correctly when the action making greater progress receives the larger multiplier. For negative credit ($d = -1$), the ordering is reversed: the action making less progress, or equivalently causing greater harm, should receive the larger multiplier. Ties in α contribute 0.5, and an uninformative ordering therefore has an expected accuracy of 0.5. Groups for which $\mathcal{P}^d(\tau) = \emptyset$ are excluded. This metric is a concordance measure restricted to the within-trajectory, same-direction comparisons over which the redistribution in Eq. 16 operates.

WebShop: page-level score. Following Xiong et al. (Xiong et al., 2024), we extend the product-level scoring function of WebShop (Yao et al., 2022) to obtain a progress score for intermediate pages. Let $r(p, c)$ denote the reward for purchasing product p with option combination c under the task goal, evaluated using the product's first listed price. For the page reached after each action, we define the score S as follows:

- *Search-results page*: the maximum, over all displayed products, of the mean $r(p, c)$ across their option combinations c. For products with more than $4{,}096$ combinations, we use a random sample of $4{,}096$.
- *Item page*: the maximum $r(p, c)$ over option combinations consistent with all option values selected so far.
- *Purchase*: the reward returned by the environment.

The initial search page is assigned score 0. Invalid actions and actions that open description, feature, or review sub-pages, or return to the search box, leave the score unchanged. Action progress is then

$$\Delta_t = S_{t+1} - S_t. \tag{34}$$

We obtain these scores by replaying each recorded task and action sequence.

ALFWorld: distance to the goal. Each ALFWorld task (Shridhar et al., 2021) is also represented as a PDDL problem in TextWorld (Côté et al., 2018). We replay the recorded action sequence in the corresponding PDDL environment and, after each action, query the planner for a shortest action sequence from the current state to the task goal. Let $L(s)$ denote the resulting plan length, with $L(s) = 0$ once the goal is satisfied. We define action progress as

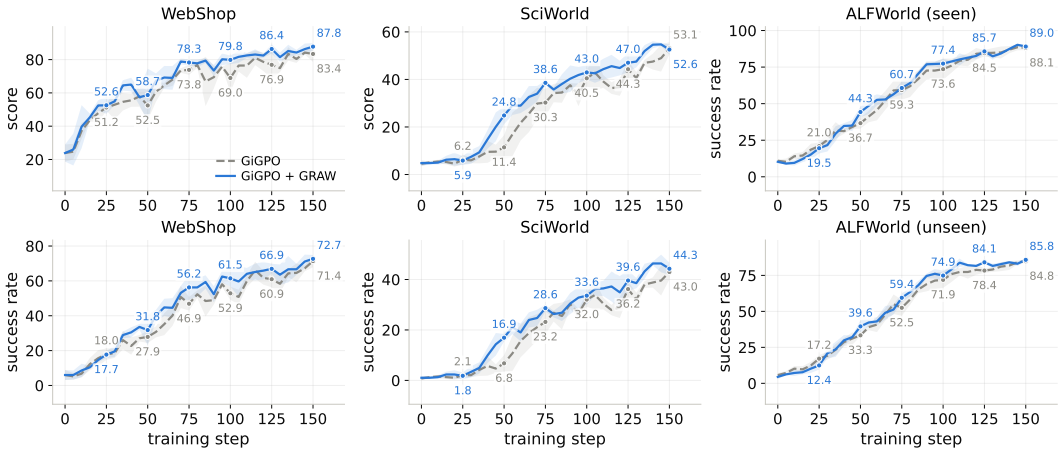
$$\Delta_t = L(s_t) - L(s_{t+1}). \tag{35}$$

Thus, $\Delta_t > 0$ indicates that the action moves the state closer to the goal, whereas $\Delta_t < 0$ indicates that it moves the state farther away. For example, putting down a required object or cooling an object that must remain heated can produce negative progress. Because the planner has access to the full symbolic state, including objects not yet observed by the agent, information-gathering actions that do not change the symbolic distance to the goal receive no positive progress.

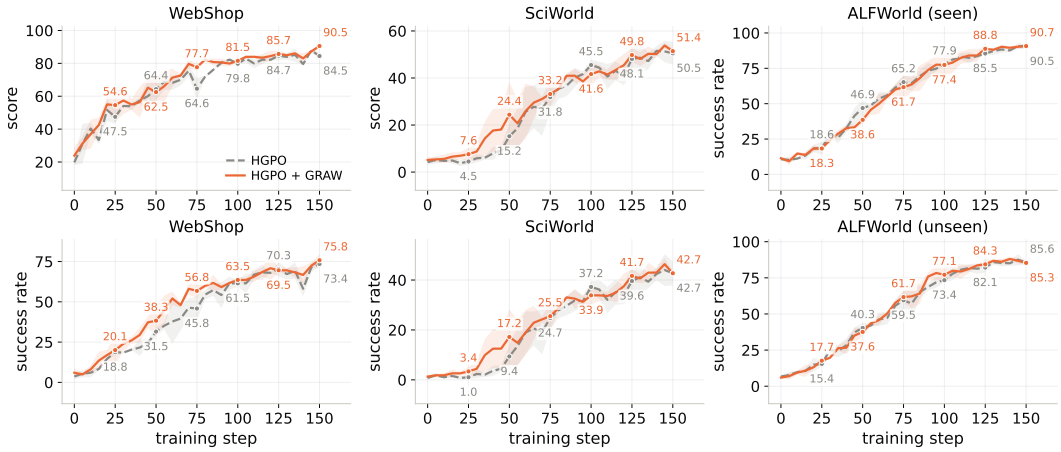
SciWorld: subgoal score. SciWorld (Wang et al., 2022) provides a task score from 0 to 100 that increases as task-specific subgoals are completed, while task-critical mistakes can produce a negative score and terminate the episode. We clip negative scores at zero and normalize the resulting progress potential as

$$\Phi(s) = \frac{\max\big(0, \mathrm{score}(s)\big)}{100}, \qquad \Delta_t = \Phi(s_{t+1}) - \Phi(s_t). \tag{36}$$

For each recorded trajectory, we recover the corresponding task variation from the task description and initial observation and replay the original action sequence. Because ScienceWorld does not guarantee a stable ordering of objects in textual observations, we compare replayed and recorded observations as unordered sets of lines. Across checkpoints, at most 3.1% of replayed observations differ from their recorded counterparts, while the final replayed outcome agrees with the recorded outcome for 96.9%–100% of trajectories.



(a) GiGPO as base estimator



(b) HGPO as base estimator

Figure 5: **Training dynamics across all benchmarks.** Test performance is evaluated every five policy-update steps on WebShop, SciWorld, and ALFWorld using Qwen2.5-1.5B-Instruct. Results are shown for GiGPO and HGPO with and without GRAW. Curves denote the mean over three random seeds, and shaded regions indicate the corresponding standard deviation.

C Additional Experiments

C.1 Training Dynamics Across Benchmarks

Figure 5 shows that the improvements persist throughout training rather than arising from a single selected checkpoint. GRAW generally improves both score and success rate for GiGPO and HGPO across evaluation steps, suggesting that the gains from relative-value-guided redistribution are sustained throughout optimization.

Table 4: **Full results across model scales.** Results for Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct across all evaluation metrics. Values in parentheses denote improvements over the corresponding base estimator.

Method	WebShop		SciWorld		ALFWorld	
	Score	Succ	Score	Succ	Seen	Unseen
Qwen2.5-3B-Instruct						
GiGPO	87.66	76.56	50.37	42.19	88.57	90.30
+ GRAW	88.85$_{(+1.19)}$	77.34$_{(+0.78)}$	55.53$_{(+5.16)}$	46.09$_{(+3.90)}$	90.71$_{(+2.14)}$	91.79$_{(+1.49)}$
HGPO	87.24	74.22	54.56	46.09	85.71	91.04
+ GRAW	90.00$_{(+2.76)}$	75.78$_{(+1.56)}$	57.38$_{(+2.82)}$	46.88$_{(+0.79)}$	90.00$_{(+4.29)}$	91.04$_{(+0.00)}$
Qwen2.5-7B-Instruct						
GiGPO	88.66	80.47	51.90	42.97	90.00	91.79
+ GRAW	91.21$_{(+2.55)}$	82.03$_{(+1.56)}$	56.68$_{(+4.78)}$	46.09$_{(+3.12)}$	95.00$_{(+5.00)}$	94.78$_{(+2.99)}$
HGPO	89.62	81.25	53.63	42.97	93.57	93.28
+ GRAW	92.18$_{(+2.56)}$	81.25$_{(+0.00)}$	63.37$_{(+9.74)}$	52.34$_{(+9.37)}$	96.43$_{(+2.86)}$	94.03$_{(+0.75)}$

C.2 Computational Efficiency

Parametric credit estimators typically introduce substantial training overhead because they require an additional learned model to score states or actions. GRAW avoids this overhead by construction. Rather than introducing a second model, it attaches only a lightweight two-layer value head ($H \to H/4 \to 1$) to the policy representation. The head reuses hidden states already produced during the standard policy forward pass, requiring no additional backbone inference. Credit redistribution itself operates only on the computed advantages through a per-trajectory softmax and therefore adds negligible computation.

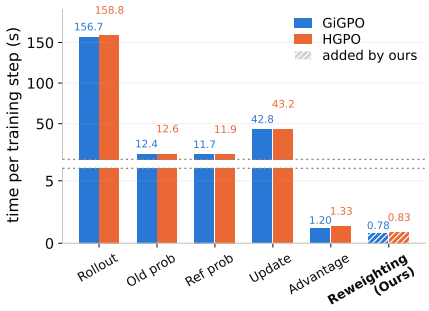


Figure 6: **Computational efficiency of GRAW.** Task-averaged wall-clock breakdown per training step on ALFWorld using Qwen2.5-1.5B-Instruct on two H200 GPUs.

As shown in Figure 6, GRAW introduces only minor additional computation over its non-parametric base estimators, while methods with a separate learned credit model incur substantially larger training overhead. By reusing policy hidden states through a lightweight value head, GRAW requires neither an additional backbone forward pass nor extra rollouts or environment interactions. It therefore incorporates learned relative action ranking while largely preserving the computational efficiency of non-parametric step-level RL.

C.3 Full Results Across Model Scales

Table 4 reports the full evaluation metrics for the model-scale experiments in Section 5.3. We additionally report the full metric set: score and success rate for WebShop and SciWorld, and seen and unseen success rates for ALFWorld. Across both Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct, GRAW improves GiGPO and HGPO on nearly all metrics, confirming that the gains observed in the main comparison are not specific to the selected metrics. The improvements are particularly pronounced on SciWorld at 7B, while performance is preserved even on metrics where the corresponding base estimator is already strong.

C.4 Qualitative Analysis of Credit Redistribution

Failed WebShop trajectory. Figure 7 illustrates how GRAW redistributes credit within a failed trajectory. The initial search is reasonable: it retrieves candidate products relevant to the requested attributes. The critical error occurs at Action 2, where the agent selects an incorrect product. Actions 3 and 4 subsequently choose the requested gender and size, whereas Action 5 commits another error by purchasing the product before selecting the required black color.

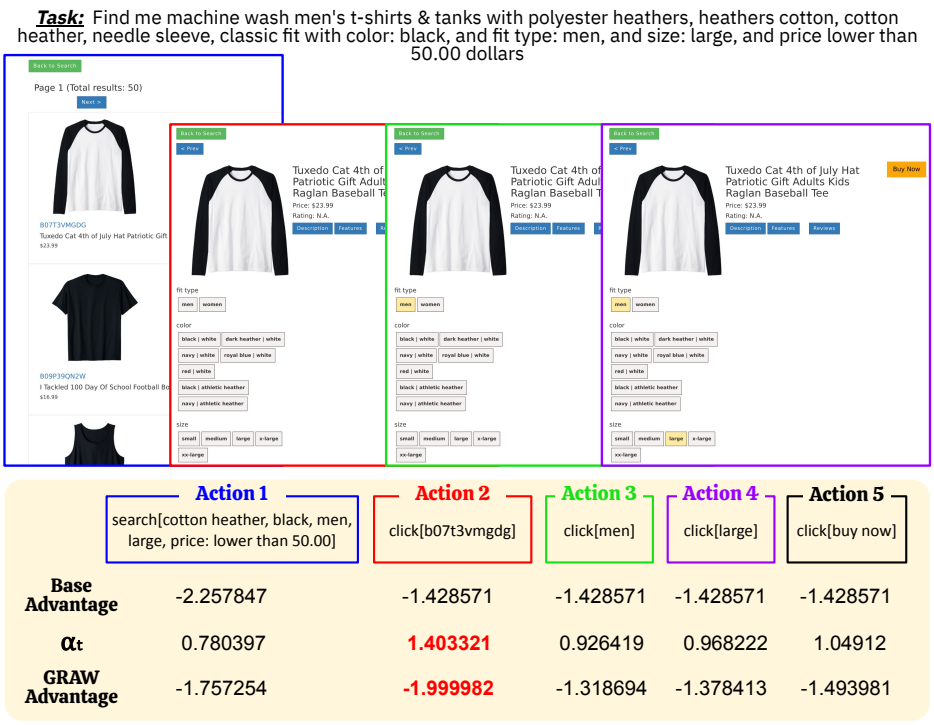


Figure 7: **Qualitative example of credit redistribution on a failed WebShop trajectory.** The base GiGPO advantage assigns the strongest negative credit to the initial search and identical negative credit to Actions 2–5. However, the search itself retrieves relevant candidates, whereas Action 2 selects an incorrect product and Action 5 purchases it before specifying the required color. GRAW preserves the negative update direction but redistributes its magnitude according to the learned multiplier α_t: Action 2 becomes the most strongly penalized action, and Action 5 receives stronger negative credit than Actions 3 and 4.

The base GiGPO credit does not reflect this distinction. It assigns the largest negative advantage to the initial search (-2.26), while Actions 2–5 all receive the same advantage (-1.43). In contrast, GRAW preserves the negative direction but reallocates its magnitude according to relative action contribution. Action 2 receives the largest multiplier ($\alpha_t = 1.40$) and consequently the strongest negative advantage (-2.00), while the premature purchase at Action 5 is also penalized more strongly than the intermediate attribute-selection actions. This example illustrates the intended behavior of GRAW: the learned signal does not change which actions are negatively credited, but differentiates *how strongly* they should be suppressed according to their relative contribution to failure.

Successful ALFWorld trajectory. Figure 8 shows the complementary behavior in a successful trajectory. Although GiGPO assigns positive credit to all actions, the magnitudes do not necessarily reflect which steps make the most direct progress toward task completion. GRAW selectively amplifies actions that are more consequential to completing the trajectory. In particular, `go to sinkbasin 1` reduces the PDDL distance-to-goal from 4 to 3 and receives $\alpha_t = 1.24$, increasing its advantage from 1.59 to 1.97. The subsequent `take potato 2 from sinkbasin 1` further reduces the distance from 3 to 2 and is also upweighted with $\alpha_t = 1.15$. These actions establish the key intermediate state needed for the remaining execution steps, and GRAW assigns them greater positive credit accordingly. This example illustrates that credit redistribution is useful not only for identifying consequential errors in failed trajectories, but also for emphasizing actions that make meaningful progress in successful ones.

Task: Place a potato in the microwave.



	Action 1 go to fridge 1	Action 2 go to microwave 1	Action 3 open microwave 1	Action 4 go to fridge 1	Action 5 open fridge 1	Action 6 go to sinkbasin 1	Action 7 take potato 2 from sinkbasin 1	Action 8 go to microwave 1	Action 9 move potato 2 to microwave 1
Base Advantage	1.437798	1.288256	1.069505	1.09372	1.872322	1.59076	0.70906	0.449834	1.185892
α_t	1.252698	0.957359	0.65107	0.903268	0.865691	**1.239522**	**1.154526**	0.918736	0.973033
GRAW Advantage	1.801127	1.233324	0.696322	0.987922	1.620854	**1.971781**	**0.818628**	0.413278	1.153911

Figure 8: **Qualitative example of credit redistribution on a successful ALFWorld trajectory.** The base GiGPO advantages assign positive credit throughout the successful trajectory, but do not directly reflect which actions make decisive progress toward completion. GRAW redistributes this credit according to learned relative value movement. In particular, the task-critical steps of navigating to the sink basin and taking the required object receive enlarged multipliers ($\alpha_t = 1.24$ and 1.15), increasing their positive advantages from 1.59 to 1.97 and from 0.71 to 0.82, respectively.

C.5 Ablation on HGPO

We repeat the ablation in Section 5.5 using HGPO as the base estimator. Figure 9 shows the same trends observed with GiGPO. Replacing direction-preserving redistribution with additive integration substantially degrades performance, particularly on ALFWorld, again highlighting the importance of preserving the base update direction. Removing grouped targets still improves over the HGPO base, confirming that relative-value-guided redistribution remains useful even with single-continuation supervision, while grouped targets provide an additional benefit. Finally, a separate critic yields the strongest overall performance, reproducing the same accuracy–computation trade-off observed with GiGPO. These results show that the contributions of GRAW are consistent across different non-parametric base estimators.

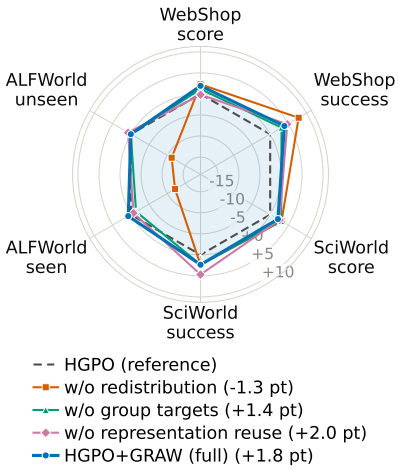


Figure 9: **Ablation of GRAW on HGPO.**

D TASK PROMPTS

Following Feng et al. (2025) and He et al. (2026a), we use identical instruction prompts for all baselines and our method. For reproducibility, we reproduce the prompts used for the three benchmarks verbatim below.

D.1 WEBSHOP

You are an expert autonomous agent operating in the WebShop e−commerce environment.
Your task is to: {task_description}.
Prior to this step, you have already taken {step_count} step(s).
Below are the most recent {history_length} observations and the corresponding actions you took:
{action_history}
You are now at step {current_step} and your current observation is: {current_observation}.
Your admissible actions of the current situation are: [{available_actions}].

Now it's your turn to take one action for the current step.
You should first reason step−by−step about the current situation, then think carefully which admissible action best advances the shopping goal.
This reasoning process MUST be enclosed within <think> </think> tags.
Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

D.2 SCIWORLD

You are an expert autonomous agent operating in the ScienceWorld environment, which is a text−based virtual environment centered around accomplishing tasks from the elementary science curriculum.
Your task is to: {task_description}.
Prior to this step, you have already taken {step_count} step(s).
Below are the most recent {history_length} observations and the corresponding actions you took:
{action_history}
You are now at step {current_step} and your current observation is: {current_observation}.
Your admissible actions of the current situation are: [{available_actions}].

Now it's your turn to take one action for the current step.
You should first reason step−by−step about the current situation, then think carefully which admissible action best advances the task goal.
This reasoning process MUST be enclosed within <think> </think> tags.
Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

D.3 ALFWORLD

You are an expert agent operating in the ALFRED Embodied Environment.
Your task is to: {task_description}.
Prior to this step, you have already taken {step_count} step(s).
Below are the most recent {history_length} observations and the corresponding actions you took:
{action_history}
You are now at step {current_step} and your current observation is: {current_observation}.
Your admissible actions of the current situation are: [{admissible_actions}].

Now it's your turn to take an action.
You should first reason step−by−step about the current situation.
This reasoning process MUST be enclosed within <think> </think> tags.
Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Algorithm 1 GRAW: Direction-Preserving Credit Redistribution

Require: Policy π_θ; relative-value head f_ϕ; non-parametric base estimator $\mathcal{B}$; discount γ; redistribution schedule $\{\beta_k\}_{k=1}^{K}$; training iterations K

for $k = 1, 2, \ldots, K$ **do**
  % Stage 1: Rollout collection
  Sample rollout groups with the current policy π_θ and collect trajectories $\{\tau\}$
  % Stage 2: Pre-update value scoring
  Compute old action log-probabilities and retain policy hidden states $\{h_t\}$
  for each visited step t **do**
    $V_\phi^{\text{old}}(s_t) \leftarrow f_\phi(\text{sg}(h_t))$ (Eq. 10)
  end for
  % Stage 3: Base credit estimation
  Compute base step advantages $\{A_t^{\mathcal{B}}\} \leftarrow \mathcal{B}(\{\tau\})$
  $A_t^{GRAW} \leftarrow A_t^{\mathcal{B}}$ for all steps t
  % Stage 4: Relative-value-guided redistribution
  for each visited step t **do**
    Construct target y_t from repeated visits or Bellman recursion (Eq. 11)
  end for
  for each trajectory τ and step t **do**
    $m_t \leftarrow \gamma V_\phi^{\text{old}}(s_{t+1}) - V_\phi^{\text{old}}(s_t)$ (Eq. 13)
  end for
  $\sigma_m \leftarrow \max(\text{std}_{u\in\mathcal{D}}(m_u), \epsilon)$
  for each trajectory τ and direction $d \in \{-1, +1\}$ **do**
    $\mathcal{G}^d(\tau) \leftarrow \{t : \text{sign}(A_t^{\mathcal{B}}) = d\}$
    if $\mathcal{G}^d(\tau) \neq \emptyset$ **then**
      $M^d(\tau) \leftarrow \sum_{u\in\mathcal{G}^d(\tau)} |A_u^{\mathcal{B}}|$ (Eq. 14)
      for $t \in \mathcal{G}^d(\tau)$ **do**
        $z_t \leftarrow d\, m_t/\sigma_m$ (Eq. 15)
      end for
      $Z^d \leftarrow \sum_{u\in\mathcal{G}^d(\tau)} |A_u^{\mathcal{B}}| \exp(z_u/\beta_k)$
      for $t \in \mathcal{G}^d(\tau)$ **do**
        $\alpha_t \leftarrow \dfrac{M^d(\tau)\exp(z_t/\beta_k)}{Z^d}$
        $A_t^{GRAW} \leftarrow \alpha_t A_t^{\mathcal{B}}$ (Eq. 16)
      end for
    end if
  end for
  % Stage 5: Joint policy and value-head update
  for each minibatch **do**
    Compute the base policy loss using A_t^{GRAW} in place of $A_t^{\mathcal{B}}$
    Compute $\mathcal{L}_{\text{head}}(\phi)$ from targets $\{y_t\}$ (Eq. 12)
    Update θ and ϕ jointly with separate parameter groups
  end for
end for

D.4 Algorithm

Algorithm 1 summarizes the training procedure of GRAW on top of a non-parametric base estimator $\mathcal{B}$ over K iterations. At each iteration, the relative-value scores used for redistribution are computed with the current value head before any parameter updates. Specifically, GRAW reuses the hidden states produced during the log-probability forward pass to obtain $V_\phi^{\text{old}}(s_t)$, requiring no additional backbone forward pass. These pre-update scores are used to compute relative movements and redistribute the base advantages. Value targets are constructed separately from collected returns through repeated-state pooling and successor-target propagation, without using the head's predictions. Finally, the policy and value head are updated within the same minibatch loop, with separate parameter groups. Because the value-head input is detached, $\mathcal{L}_{\text{head}}$ updates only ϕ and does not directly modify the policy representation.